

- Determining your hardware coordinates (especially core ID)
  - DCMF\_Hardware\_t structure
  - DCMF\_Messager\_rank2torus()
- If you just want coordinates, use rank2torus. There are other pieces of information in the DCMF\_Hardware\_t structure. See dcmf.h

```
#include<stdio.h>
#include <mpi.h>
#include "dcmf.h"
int main(int argc, char *argv[])
{
    DCMF_Hardware_t hw;
    int rank;
    int x, y, z, t;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    DCMF_Hardware(&hw);

    fprintf(stderr,"I am rank %d. My coords: %d %d %d, my core: %d\n",
            rank, hw.xCoord, hw.yCoord, hw.zCoord, hw.tCoord);
    DCMF_Messager_rank2torus(rank, &x, &y, &z, &t);
    fprintf(stderr,"Rank2Torus %d: %d, %d, %d, core %d\n", rank, x, y, z, t);

    MPI_Finalize();
    return;
}
```

# Finding your compiler type/ architecture type

- `#ifdef __bg__`
- `#warning this is set for the BG machines`
- `#endif`
- `#ifdef __bgp__`
- `#warning this is set for BGP specifically`
- `#endif`
- `#ifdef __xIC__`
- `#warning this is set for XL compilers`
- `#endif`

# More useful env vars

- **BG\_MAXALIGNEXP=1**
  - App dies on first alignment exception. Default is to die on 1000 alignment exceptions.
  - Alignment exceptions hurt performance.
  - Set this to 1 and see the first misaligned piece of data
- **BG\_COREDUMPONEXIT=1**
  - Always dumps core

# Using gdb with LL

- Instead of having mpirun as your program to execute, execute ‘DISPLAY={} xterm’

```
#@ output = $(job_name).$(jobid).out
...
#@ queue
#/usr/bin/mpirun {}
export DISPLAY=mymachine:1 xterm
```

In the new xterm: /usr/bin/mpirun {} –start\_gdbserver {path to gdbserver}